Root-Weighted Tree Automata and their Applications to Tree Kernels

Ludovic Mignot, Nadia Ouali-Sebti, Djelloul Ziadi

LITIS, Université de Rouen, 76801 Saint-Étienne du Rouvray Cedex, France
{ludovic.mignot, nadia.ouali-sebti, djelloul.ziadi}@univ-rouen.fr

Abstract. In this paper, we define a new kind of weighted tree automata where the weights are only supported by final states. We show that these automata are sequentializable and we study their closures under classical regular and algebraic operations.
We then use these automata to compute the subtree kernel of two finite tree languages in an efficient way. Finally, we present some perspectives involving the root-weighted tree automata.

1 Introduction

Kernel methods have been widely used to extend the applicability of many well-known algorithms, such as the Perceptron [1], Support Vector Machines [5], or Principal Component Analysis [17] . Tree kernels are interesting approaches in areas of machine learning based natural language processing. They have been applied to reduce such effort for several natural language tasks, *e.g.* relation extraction [16], syntactic parsing re-ranking [3], named entity recognition[6,7] and Semantic Role Labeling [12].

The main idea of tree kernels is to compute the number of common substructures (subtrees and subset trees) between two trees t_1 and t_2. In [14], Moschitti defined an algorithm for the computation of this type of tree kernels which computes the kernels between two syntactic parse trees in $O(m \times n)$ time, where m and n are the number of nodes in the two trees. To do this, Moschitti modified the function proposed by Collins and Duffy in [3] by introducing a parameter $\sigma \in \{0, 1\}$ which enables the SubTrees ($\sigma = 1$) or the SubSet Trees ($\sigma = 0$) evaluation and which is defined for two trees t_1 and t_2 as follows: Given a set of substructures $\mathcal{S} = \{s_1, s_2, \ldots\}$, they defined the indicator function $I_i(n)$ which is equal to 1 if the substructure s_i is rooted at node n and 0 otherwise. They defined the tree kernel function between the two trees t_1 and t_2 as follows: $K(t_1, t_2) = \sum_{n_1 \in N_{t_1}} \sum_{n_2 \in N_{t_2}} \Delta(n_1, n_2)$ where N_{t_1} and N_{t_2} are the number of nodes in t_1 and t_2 respectively and $\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{S}|} I_i(n_1) \cdot I_i(n_2)$. We can then compute Δ as follows:

- if the productions at n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$,
- if the productions at n_1 and n_2 are the same and n_1 and n_2 are leaves then $\Delta(n_1, n_2) = 1$,
- if the productions at n_1 and n_2 are the same and n_1 and n_2 are not leaves then $\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (\sigma + \Delta(C^j_{n_1}, C^j_{n_2}))$, where $nc(n_1)$ is the number of children of n_1 and C^j_n is j^{th} child of the node n.

In [13], Moschitti proposed a new convolution kernel, namely the Partial Tree kernel, to fully exploit dependency trees. He proposed an efficient algorithm for its computation which is based on applying the selection of tree nodes with non-null kernel. In the following we propose a new technique to compute these kind of tree kernels using weighted tree automata. We will start by defining a new class of weighted tree automata that we call *rooted weighted tree automata* and we will prove some properties of these weighted tree automata. Then we will show that tree kernels can be computed efficiently using a general intersection of rooted weighted tree automata defined here.

The paper is organized as follows. In the following section, we introduce the trees, operations in trees and in tree languages and some preliminary notions used in the remaining sections. Section 3, presents the sequentialization of rooted weighted tree automata and the closure of these automata under rational or algebraic operations. In Section 4 we present an efficient computation of the subtree kernel of two finite tree series. Finally, the different results described in this paper are given in the conclusion.

2 Preliminaries

Let Σ be a graded alphabet. A *tree* t over Σ is inductively defined $t = f(t_1, \ldots, t_k)$ where k is any integer, f is any symbol in Σ_k and $t_1, \ldots, t_k$ are any k trees over Σ. We denote by T_Σ the set of trees over Σ. A *tree language* over Σ is a subset of T_Σ.

Let c be a symbol in Σ_0, L be a tree language over Σ and t be a tree in T_Σ. The *tree substitution* of c by L in t, denoted by $t_{\{c \leftarrow L\}}$, is the language inductively defined by:

- L if $t = c$;
- $\{d\}$ if $t = d \in \Sigma_0 \setminus \{c\}$;
- $f(t_{1\{c \leftarrow L\}}, \ldots, t_{k\{c \leftarrow L\}})$ if $t = f(t_1, \ldots, t_k)$ with $f \in \Sigma_k$ and $t_1, \ldots, t_k$ any k trees over Σ.

The *c-product* $L_1 \cdot_c L_2$ of two tree languages L_1 and L_2 over Σ is the tree language $L_1 \cdot_c L_2$ defined by $\bigcup_{t \in L_1} t_{\{c \leftarrow L_2\}}$. The *iterated c-product* of a tree language L over Σ is the tree language L^{n_c} recursively defined by:

- $L^{0_c} = \{c\}$,
- $L^{(n+1)_c} = L^{n_c} \cup L \cdot_c L^{n_c}$.

The *c-closure* of the tree language L is the language L^{*_c} defined by $\bigcup_{n \geq 0} L^{n_c}$.

In the following, we make use of weighted tree automata in order to compute tree kernels. See [4] for details about classical tree automata.

Let t be a tree over an alphabet Σ. The tree $t^\sharp$ is obtained by indexing the symbols of t by its position in a prefix course. We denote by $\Sigma_{t^\sharp}$ the set of the indexed symbols that appears in $t^\sharp$. The function h is the dual function, which drops the indexes ($\mathrm{h}(t^\sharp) = t$). Notice that the function h defines an equivalence relation over $T_{\Sigma_{t^\sharp}}$. Indeed, let t_1 and t_2 be two trees in $T_{\Sigma_{t^\sharp}}$. We define the relation $\sim_\mathrm{h}$ by $t_1 \sim_\mathrm{h} t_2 \Leftrightarrow \mathrm{h}(t_1) = \mathrm{h}(t_2)$. Since $\sim_\mathrm{h}$ is a relation based on the equality of images by h, it can be shown that

Lemma 1. *The relation $\sim_\mathrm{h}$ is an equivalence relation.*

Let Σ be an alphabet and $t = f(t_1, \ldots, t_k)$ be a tree in T_Σ.

The set $\mathrm{SubTree}(t)$ is the set inductively defined by $\mathrm{SubTree}(t) = \{t\} \cup \bigcup_{1 \leq j \leq k} \mathrm{SubTree}(t_j)$. Let L be a tree language over Σ. The set $\mathrm{SubTreeSet}(L)$ is the set defined by $\mathrm{SubTreeSet}(L) = \bigcup_{t \in L} \mathrm{SubTree}(t)$.

The formal tree series $\mathrm{SubTreeSeries}_t$ is the tree series over $\mathbb{N}$ inductively defined by $\mathrm{SubTreeSeries}_t = t + \sum_{1 \leq j \leq k} \mathrm{SubTreeSeries}_{t_j}$. Let L be a finite tree language. The formal tree series $\mathrm{SubTreeSeries}_L$ is the tree series over $\mathbb{N}$ defined by $\mathrm{SubTreeSeries}_L = \sum_{t' \in L} \mathrm{SubTreeSeries}_{t'}$. Let us notice that if L is not finite, since Σ is a finite set of symbols, there exists a tree t in Σ_0 that appears an infinite times as a subtree in L; thus $\mathrm{SubTreeSeries}_L$ is a tree series over $\mathbb{N} \cup \{+\infty\}$.

Definition 1. *Let L_1 and L_2 be two finite tree languages. The* subtree series kernel *of L_1 and L_2 is the integer* $\mathrm{KerSeries}(L_1, L_2)$ *defined by:*

$$\mathrm{KerSeries}(L_1, L_2) = \sum_{t \in T_\Sigma} (\mathrm{SubTreeSeries}_{L_1} \times \mathrm{SubTreeSeries}_{L_2})(t).$$

Example 1. Let Σ be the graded alphabet defined by $\Sigma_0 = \{a, b\}$, $\Sigma_1 = \{h\}$ and $\Sigma_2 = \{f\}$. Let us consider the trees $t_1 = f(h(a), f(h(a), b))$, $t_2 = f(h(a), h(b))$ and $t_3 = f(f(b, h(b)), f(h(a), h(b)))$. Then it can be shown that:

- $\mathrm{SubTree}(t_1) = \{t_1, f(h(a), b), h(a), a, b\}$
- $\mathrm{SubTree}(t_2) = \{t_2, h(a), h(b), a, b\}$
- $\mathrm{SubTreeSeries}_{t_1} = \mathbb{P}_{t_1} = t + f(h(a), b) + 2h(a) + 2a + b$
- $\mathrm{SubTreeSeries}_{t_2} = \mathbb{P}_{t_2} = t_2 + h(b) + h(a) + a + b$
- $\mathrm{SubTreeSeries}_{t_3} = \mathbb{P}_{t_3} = t_3 + f(b, h(b)) + t_2 + 2h(b) + h(a) + 3b + a$
- $\mathrm{SubTreeSeries}_{\{t_1, t_2\}} = \mathbb{P}_{\{t_1, t_2\}} = \mathbb{P}_{t_1} + \mathbb{P}_{t_2} = t + t_2 + f(h(a), b) + 3h(a) + h(b) + 3a + 2b$
- $\mathbb{P}_{\{t_1, t_2\}} \times \mathbb{P}_{\{t_3\}} = t_2 + 2h(b) + 3h(a) + 6b + 3a$
- $\mathrm{KerSeries}(\{t_1, t_2\}, \{t_3\}) = 15$

3 Tree Series and Root-Weighted Tree Automata

A *formal tree series* [2,9] $\mathbb{P}$ over a set S is a mapping from T_Σ to S. Let $\mathbb{M} = (M, +)$ be a monoid which identity is 0. The *support* of $\mathbb{P}$ is the set $\text{Support}(\mathbb{P}) = \{t \in T_\Sigma \mid \mathbb{P}(t) \neq 0\}$. Any formal tree series can be view as a formal sum $\mathbb{P} = \sum_{t \in T_\Sigma} (\mathbb{P}(t), t)$. In this case, the formal sum is considered associative and commutative.

Formal tree series can be realized by weighted tree automata. Weighted tree automata were defined over semirings [8] or multioperator monoids [10]. In this paper, we use particular automata, the weights of which belong to a monoid or a semiring, and only label the finality of states. Consequently, the automata we use are a strcit subclasses of weighted tree automata, with particular properties.

3.1 Root-Weighted Tree Automata

Definition 2. *Let $\mathbb{M} = (M, +)$ be a commutative monoid. A* $\mathbb{M}$-Root Weighted Tree Automata *($\mathbb{M}$-RWTA) is a 4-tuple (Σ, Q, ν, δ) where:*

- $\Sigma = \bigcup_{k \in \mathbb{N}} \Sigma_k$ *is a graded alphabet,*
- Q *is a finite set of* states*,*
- ν *is a function from Q to M called the* root weight function*,*
- δ *is a subset of $Q \times \Sigma_k \times Q^k$, called the* transition set*.*

When there is no ambiguity, a $\mathbb{M}$-RWTA is called a RWTA.

The root weight function ν is extended to $2^Q \to M$ for any subset S of Q by $\nu(S) = \sum_{s \in S} \nu(s)$. The function ν is equivalent to the finite subset of $Q \times M$ defined for any couple (q, m) in $Q \times M$ by $(q, m) \in \nu \Leftrightarrow \nu(q) = m$.

The transition set δ is equivalent to the function from $\Sigma_k \times Q^k$ to 2^Q defined for any symbol f in Σ_k and for any k-tuple $(q_1, \ldots, q_k)$ in Q^k by $q \in \delta(f, q_1, \ldots, q_k) \Leftrightarrow (q, f, q_1, \ldots, q_k) \in \delta$. The function δ is extended to $\Sigma_k \times (2^Q)^k \to 2^Q$ as follows: for any symbol f in Σ_k, for any k-tuple $(Q_1, \ldots, Q_k)$ of subsets of Q, $\delta(f, Q_1, \ldots, Q_k) = \bigcup_{(q_1, \ldots, q_k) \in Q_1 \times \cdots \times Q_k} \delta(f, q_1, \ldots, q_k)$. Finally, the function Δ is the function from T_Σ to 2^Q defined for any tree $t = f(t_1, \ldots, t_k)$ in T_Σ by $\Delta(t) = \delta(f, \Delta(t_1), \ldots, \Delta(t_k))$.

A *weight* of a tree associated with A is $\nu(\Delta(t))$. The *formal tree series realized* by A is the formal tree series over M denoted by $\mathbb{P}_A$ and defined by $\mathbb{P}_A(t) = \nu(\Delta(t))$, with $\nu(\emptyset) = 0$ with 0 the identity of $\mathbb{M}$.

Example 2. Let us consider the graded alphabet Σ defined by $\Sigma_0 = \{a\}$, $\Sigma_1 = \{h\}$ and $\Sigma_2 = \{f\}$. Let $\mathbb{M} = (\mathbb{N}, +)$. The RWTA $A = (\Sigma, Q, \nu, \delta)$ defined by

- $Q = \{1, 2, 3, 4, 5\}$,
- $\nu = \{(1, 0), (2, 3), (3, 1), (4, 2), (5, 4)\}$,
- $\delta = \{(1, a), (3, a)(2, f, 1, 3), (4, f, 3, 3), (5, h, 2), (5, h, 4), (5, h, 5)\}$,

is represented in Figure 1 and realized the tree series:

$$\mathbb{P}_A = a + 5f(a, a) + 4h(f(a, a)) + 4h(h(f(a, a))) + \cdots + 4h(h(\ldots h(f(a, a))\ldots)) + \cdots.$$

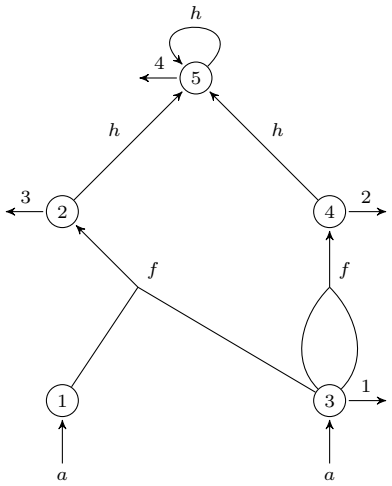

Fig. 1. The RWTA A.

Let $A_1 = (\Sigma, Q_1, \nu_1, \delta_1)$ and $A_2 = (\Gamma, Q_2, \nu_2, \delta_2)$ be two RWTAs. A function μ is a *morphism of RWTA* from A_1 to A_2 if:

- $\forall q \in Q_1$, $\mu(q) \in Q_2$,
- $\forall f \in \Sigma_k$, $\mu(f) \in \Gamma_k$,
- $\forall (q, f, q_1, \ldots, q_k) \in \delta_1$, $(\mu(q), \mu(f), \mu(q_1), \ldots, \mu(q_k)) \in \delta_2$,
- $\forall q \in Q_1$, $\nu_2(\mu(q)) = \nu_1(q)$.

A morphism μ from A_1 to A_2 is said to be an *isomorphism* if there exists a morphism μ^{-1} from A_2 to A_1. In this case, A_1 and A_2 are said to be *isomorphic*. It can be shown by induction over the structure of any tree t in T_Σ that if A_1 and A_2 are isomorphic w.r.t. a morphism μ then $\Delta_2(\mu(t)) = \bigcup_{q \in \Delta_1(t)} \{\mu(q)\}$. Therefore

Lemma 2. *Let A_1 be a RWTA over an alphabet Σ. Let A_2 be a RWTA isomorphic to A_1 w.r.t. a morphism μ. Then for any tree t in T_Σ,*

$$\mathbb{P}_{A_1}(t) = \mathbb{P}_{A_2}(\mu(t)).$$

As a direct corollary, it holds

Corollary 1. *Two isomorphic RWTAs over the same alphabet realize the same tree series.*

3.2 RWTA Sequentialization

The RTWA A is said to be *sequential* if and only if for any tree t in T_Σ, $\mathrm{Card}(\Delta(t)) \leq 1$. Unlike the case of classical weighted tree and word automata, the RWTAs are sequentializable.

Theorem 1. *For any RTWA A, there exists a sequential RTWA A' such that $\mathbb{P}_A = \mathbb{P}_{A'}$.*

In order to prove Theorem 1, let us define the subset construction [15] for any RWTA.

Definition 3. *Let $A = (\Sigma, Q, \nu, \delta)$ be a RWTA. The* sequential RWTA associated with *A is the RTWA $A' = (\Sigma, 2^Q, \nu', \delta')$ defined by:*

- $\forall S \subset Q$, $\nu'(S) = \sum_{s \in S} \nu(s)$;
- $\forall f \in \Sigma_k$, $\forall Q_1, \ldots, Q_k \subset Q$, $\delta'(f, Q_1, \ldots, Q_n) = \{\delta(f, Q_1, \ldots, Q_k)\}$.

Notice that ν' is equal to the extension of ν over the subsets of Q. However, δ' is not equal to the extension of δ over sets since it necessarily returns a singleton.

Lemma 3. *Let $A = (\Sigma, Q, \nu, \delta)$. Let $A' = (\Sigma, 2^Q, \nu', \delta')$ be the sequential RWTA associated with A. For any tree t in T_Σ,*

$$\Delta'(t) = \{\Delta(t)\}.$$

Proof. By definition of Δ', $\Delta'(f(t_1, \ldots, t_k)) = \delta'(f, \Delta'(t_1), \ldots, \Delta'(t_k))$.

1. If $k = 0$, then $\Delta'(f) = \delta'(f)$. Moreover, by definition of A', $\delta'(f) = \{\delta(f)\}$. Since by definition of Δ, $\delta(f) = \Delta(f)$, it holds that $\Delta'(f) = \{\Delta(f)\}$.
2. Suppose that $k \neq 0$. According to induction hypothesis, it holds that $\Delta'(f(t_1, \ldots, t_k)) = \delta'(f, \{\Delta(t_1)\}, \ldots, \{\Delta(t_k)\})$. By definition of δ', $\delta'(f, \{\Delta(t_1)\}, \ldots, \{\Delta(t_k)\}) = \{\delta(f, \Delta(t_1), \ldots, \Delta(t_k))\}$, that equals by definition $\{\Delta(f(t_1, \ldots, t_k))\}$. Hence $\Delta'(t) = \{\Delta(t)\}$.

■

Proposition 1. *Let A be a RWTA and A' be the sequential RWTA associated with A. Then:*
A' is a sequential RTWA that realizes $\mathbb{P}_A$.

Proof. Let $A = (\Sigma, Q, \nu, \delta)$ and $A' = (\Sigma, 2^Q, \nu', \delta')$. Let $t = f(t_1, \ldots, t_k)$ be a tree in Σ. According to Lemma 3, $\Delta'(t) = \{\Delta(t)\}$. As a direct consequence, $\mathrm{Card}(\Delta'(t)) = 1$ (since the state $\emptyset$ may be reached) and $\mathbb{P}_{A'}(t) = \nu'(\Delta'(t)) = \nu'(\Delta(t)) = \mathbb{P}_A(t)$. Hence A' is a sequential RTWA that realizes $\mathbb{P}_A$. ■

Example 3. Let us consider the RWTA defined in Example 2. The sequential RWTA associated with A is represented in Figure 2.

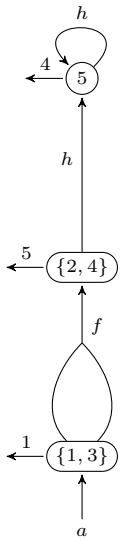

Fig. 2. The sequential RWTA associated with A.

Since a sequential RWTA is a RTWA, the set of tree series realized by a RTWA is closed under sequentialization, whatever the set of weights is. Let us now show that this set is also closed under several algebraic operations.

3.3 Sum and Product Closures

If $(M, +)$ is a commutative monoid, then the set of tree series over M realized by a RTWA is closed under the sum.

Definition 4. *Let $A_1 = (\Sigma, Q_1, \nu_1, \delta_1)$ and $A_2 = (\Sigma, Q_2, \nu_2, \delta_2)$ be two RWTAs such that $Q_1 \cap Q_2 = \emptyset$. The RWTA $A_1 + A_2$ is the RWTA $A' = (\Sigma, Q_1 \cup Q_2, \nu', \delta_1 \cup \delta_2)$ where ν' is the function defined for any state q in $Q_1 \cup Q_2$ by:*

$$\nu'(q) = \begin{cases} \nu_1(q) & \text{if } q \in Q_1, \\ \nu_2(q) & \text{otherwise,} \end{cases}$$

Notice that if Q_1 and Q_2 are not disjoint, then Q_2 can be changed using an isomorphism.

Proposition 2. *Let A_1 and A_2 be two RWTAs. Then for any tree t in T_Σ:*

$$\mathbb{P}_{A_1+A_2}(t) = \mathbb{P}_{A_1}(t) + \mathbb{P}_{A_2}(t).$$

Proof. Let $A_1 = (\Sigma, Q_1, \nu_1, \delta_1)$, $A_2 = (\Sigma, Q_2, \nu_2, \delta_2)$ and $A' = A_1 + A_2 = (\Sigma, Q', \nu', \delta')$. Let $t = f(t_1, \ldots, t_k)$ be a tree in T_Σ. Let us first show by induction over the structure of t that $\Delta'(t) = \Delta_1(t) \cup \Delta_2(t)$. By definition, $\Delta'(f(t_1, \ldots, t_k)) = \delta'(f, \Delta'(t_1), \ldots, \Delta'(t_k))$.

1. if $k = 0$, then $\Delta'(f) = \delta'(f)$. By definition of A', $\delta'(t) = \delta_1(t) \cup \delta_2(t)$ that equals by definition to $\Delta_1(f) \cup \Delta_2(f)$. Hence $\Delta'(t) = \Delta_1(t) \cup \Delta_2(t)$.
2. If $k \neq 0$, then by induction hypothesis, $\Delta'(f(t_1, \ldots, t_k)) = \delta'(f, \Delta_1(t_1) \cup \Delta_2(t_1), \ldots, \Delta_1(t_k) \cup \Delta_2(t_k))$. Since Q_1 and Q_2 are disjoint, there is no transition $(q, f, q_1, \ldots, q_n)$ in Δ' such that there exists two integers i and j such that $q_i \in Q_1$ and $q_j \in Q_2$. Therefore $\delta'(f, \Delta_1(t_1) \cup \Delta_2(t_1), \ldots, \Delta_1(t_k) \cup \Delta_2(t_k)) = \delta_1(f, \Delta_1(t_1), \ldots, \Delta_1(t_k)) \cup \delta_2(f, \Delta_2(t_1), \ldots, \Delta_2(t_k))$, that is equal to $\Delta_1(f(t_1, \ldots, t_k)) \cup \Delta_2(f(t_1, \ldots, t_k))$. Hence $\Delta'(t) = \Delta_1(t) \cup \Delta_2(t)$.

As a direct consequence, $\mathbb{P}_{A'}(t) = \nu'(\Delta_1(t) \cup \Delta_2(t)) = \nu_1(\Delta_1(t)) + \nu_2(\Delta_2(t)) = \mathbb{P}_{A_1}(t) + \mathbb{P}_{A_2}(t)$. ∎

A *semiring* is a 5-tuple $\mathbb{K} = (K, +, \times, 0, 1)$ such that:

- $(K, +)$ is a commutative monoid the identity of which is 0,
- $(K, *)$ is a monoid the identity of which is 1,
- $0 \times \alpha = \alpha \times 0 = 0$ for any α in K,
- $\times$ distributes over $+$.

In the following, we consider trees over an alphabet Σ and tree series over the semiring $\mathbb{K}$. From this structure, another stable operation can be defined for formal tree series over K.

Let $\mathbb{P}_1$ and $\mathbb{P}_2$ be two tree series. The *product of* $\mathbb{P}_1$ and $\mathbb{P}_2$ is the series $\mathbb{P}_1 \times \mathbb{P}_2$ defined for any tree t by $\mathbb{P}_1 \times \mathbb{P}_2(t) = \mathbb{P}_1(t) \times \mathbb{P}_2(t)$. Let us show now that the product can be performed *via* RWTAs.

Definition 5. *Let $A_1 = (\Sigma, Q_1, \nu_1, \delta_1)$ and $A_2 = (\Sigma, Q_2, \nu_2, \delta_2)$ be two RWTAs. The RWTA $A_1 \times A_2$ is the RWTA $A' = (\Sigma, Q' = Q_1 \times Q_2, \nu', \delta')$ defined by:*

- $\forall f \in \Sigma_k, \forall q_1 = (q_{1_1}, q_{2_1}), \ldots, q_k = (q_{1_k}, q_{2_k}) \in Q'$, $\delta'(f, q_1, \ldots, q_k) = \delta_1(f, q_{1_1}, \ldots, q_{1_k}) \times \delta_2(f, q_{2_1}, \ldots, q_{2_k})$,
- $\forall q = (q_1, q_2) \in Q'$, $\nu'(q) = \nu_1(q_1) \times \nu_2(q_2)$.

Lemma 4. *Let $A_1 = (\Sigma, Q_1, \nu_1, \delta_1)$ and $A_2 = (\Sigma, Q_2, \nu_2, \delta_2)$ be two RWTAs. Then for any tree t in T_Σ:*

$$\Delta'(t) = \Delta_1(t) \times \Delta_2(t).$$

Proof. By induction over the structure of $t = f(t_1, \ldots, t_k)$. By definition, $\Delta'(f(t_1, \ldots, t_k)) = \delta'(f, \Delta'(t_1), \ldots, \Delta'(t_k))$.

1. if $k = 0$, then $\Delta'(f) = \delta'(f)$. By definition of A', $\delta'(t) = \delta_1(t) \times \delta_2(t)$ that equals by definition to $\Delta_1(f) \times \Delta_2(f)$. Hence $\Delta'(t) = \Delta_1(t) \times \Delta_2(t)$.
2. If $k \neq 0$, then by induction hypothesis, $\Delta'(f(t_1, \ldots, t_k)) = \delta'(f, \Delta_1(t_1) \times \Delta_2(t_1), \ldots, \Delta_1(t_k) \times \Delta_2(t_k))$. According to the definition of δ',
   $\delta'(f, \Delta_1(t_1) \times \Delta_2(t_1), \ldots, \Delta_1(t_k) \times \Delta_2(t_k)) = \bigcup_{q_j = (q_{1_j}, q_{2_j}) \in \Delta_1(t_j) \times \Delta_2(t_j), 1 \leq j \leq k} \delta'(f, q_1, \ldots, q_k)$.
   By definition of A', $\delta'(f, q_1, \ldots, q_k) = \delta_1(f, q_{1_1}, \ldots, q_{1_k}) \times \delta_2(f, q_{2_1}, \ldots, q_{2_k})$, for any $q_j = (q_{1_j}, q_{2_j}) \in \Delta_1(t_j) \times \Delta_2(t_j), 1 \leq j \leq k$. Furthermore, by definition of the cartesian product of set,
   $$\begin{aligned} \Delta'(t) &= \bigcup_{q_j = (q_{1_j}, q_{2_j}) \in \Delta_1(t_j) \times \Delta_2(t_j), 1 \leq j \leq k} \delta_1(f, q_{1_1}, \ldots, q_{1_k}) \times \delta_2(f, q_{2_1}, \ldots, q_{2_k}) \\ &= \bigcup_{q_j \in \Delta_1(t_j), 1 \leq j \leq k} \delta_1(f, q_1, \ldots, q_k) \times \bigcup_{q_j \in \Delta_2(t_j), 1 \leq j \leq k) \delta_2(f, q_1, \ldots, q_k}) \end{aligned}$$
   that is equal to $\delta_1(f, \Delta_1(t_1), \ldots, \Delta_1(t_k)) \times \delta_2(f, \Delta_2(t_1), \ldots, \Delta_2(t_k)) = \Delta_1(t) \times \Delta_2(t)$ by definition.

■

Proposition 3. *Let A_1 and A_2 be two RWTAs. Then for any tree t in T_Σ:*

$$\mathbb{P}_{A_1 \times A_2}(t) = \mathbb{P}_{A_1}(t) \times \mathbb{P}_{A_2}(t).$$

Proof. Let $A_1 = (\Sigma, Q_1, \nu_1, \delta_1)$, $A_2 = (\Sigma, Q_2, \nu_2, \delta_2)$ and $A' = A_1 \times A_2 = (\Sigma, Q', \nu', \delta')$. Let $t = f(t_1, \ldots, t_k)$ be a tree in T_Σ. From Lemma 4, $\Delta'(t) = \Delta_1(t) \times \Delta_2(t)$. Hence $\mathbb{P}_{A'}(t) = \nu'(\Delta_1(t) \times \Delta_2(t))$. By definition of ν', $\mathbb{P}_{A'}(t) = \sum_{q_1 \in \Delta_1(t), q_2 \in \Delta_2(t)} \nu_1(q_1) \times \nu_2(q_2)$. Since $\mathbb{K}$ is a semiring, by distibutivity, $\mathbb{P}_{A'}(t) = (\sum_{q_1 \in \Delta_1(t)} \nu_1(q_1)) \times (\sum_{q_2 \in \Delta_2(t)} \nu_2(q_2))$. Therefore, according to the definition of Δ_1 and Δ_2), $\mathbb{P}_{A'}(t) = \nu_1(\Delta_1(t)) \times \nu_2(\Delta_2(t))$. ■

Example 4. Let us consider the RWTA A defined in Example 2 and let A' be the RWTA represented in Figure 3. The sum $A + A'$ is represented by the juxtaposition of Figure 1 and Figure 3 and the product $A \times A'$ is represented in Figure 4.

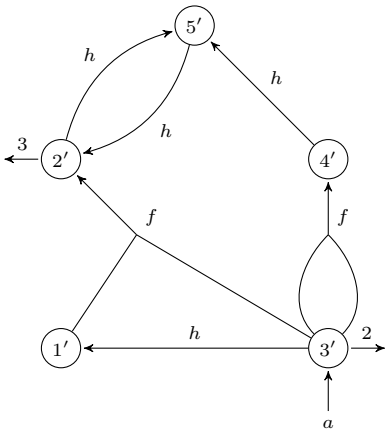

Fig. 3. The RWTA A'.

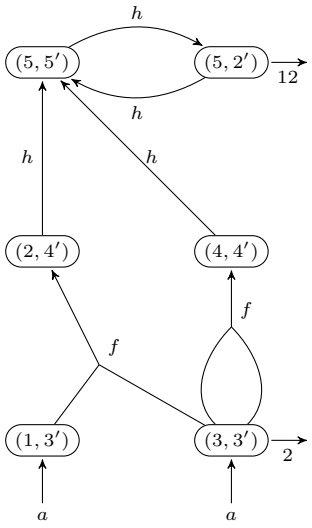

Fig. 4. The RWTA $A \times A'$.

Notice that series realized by RWTAs are not necessarily closed under classical regular operations.

3.4 Case of the *a*-Product

Let a be a symbol in Σ_0. The *a-product of* $\mathbb{P}_1$ and $\mathbb{P}_2$ is the series $\mathbb{P}_1 \cdot_a \mathbb{P}_2$ defined for any tree t by $\mathbb{P}_1 \cdot_a \mathbb{P}_2(t) = \sum_{t_1, t_2 \in T_\Sigma, t = t_1 \cdot_a t_2} \nu_1(t_1) \times \nu_2(t_2)$.

Let us show that the a-product of two series realized by some RTWAs may not be realized by any RTWA.

The *image* of a tree series $\mathbb{P}$ is the set $\mathrm{Im}(\mathbb{P}) = \{\alpha \in K \mid \exists t \in T_\Sigma, \mathbb{P}(t) = \alpha\}$.

Lemma 5. *Let A be a RWTA. Then:*

$$\mathrm{Im}(\mathbb{P}_A) \text{ is a finite set.}$$

Proof. Let $A = (\Sigma, Q, \nu, \delta)$. By definition, for any tree t in T_Σ, $\mathbb{P}_A(t) = \sum_{q \in \Delta(t)} \nu(q)$. Consequently, $\mathbb{P}_A(t)$ belongs to the subset $\{\alpha \in K \mid \exists S \subset Q, \alpha = \nu(S)\}$ of K. Therefore, $\mathrm{Card}(\mathrm{Im}(\mathbb{P}_A))$ is less than $2^{\mathrm{Card}(Q)}$.

Proposition 4. *Let Σ be an alphabet and a be a symbol in Σ_0. There exist formal tree series $\mathbb{P}_1$ and $\mathbb{P}_2$ such that $\mathrm{Im}(\mathbb{P}_1 \cdot_a \mathbb{P}_2)$ is not finite.*

Proof. Let $\mathbb{K} = (\mathbb{N}, +, \times, 0, 1)$. Let us consider the alphabet Σ defined by $\Sigma_0 = \{a, b\}$, $\Sigma_2 = \{f\}$. Let us consider the tree language L defined by $(f(a, b))^{*_a}$. Let us consider the series $\mathbb{P}_1$ and $\mathbb{P}_2$ defined for any tree t in T_Σ as follows:

– $\mathbb{P}_1(t) = \begin{cases} 1 & \text{if } t \in L, \\ 0 & \text{otherwise;} \end{cases}$
– $\mathbb{P}_2(t) = \begin{cases} 1 & \text{if } t = a, \\ 0 & \text{otherwise;} \end{cases}$

Let $A_1 = (\Sigma, Q_1, \nu_1, \delta_1)$ be the RTWA defined by:

– $Q_1 = \{0, 1\}$,
– $\nu_1(0) = 1$, $\nu_1(1) = 0$,
– $\delta_1(a) = \{0\}$, $\delta_1(b) = \{1\}$, $\delta_1(f, 0, 1) = \{f\}$.

Let $A_2 = (\Sigma, Q_2, \nu_2, \delta_2)$ be the RTWA defined by:

– $Q_2 = \{0\}$,
– $\nu_2(0) = 2$,
– $\delta_2(a) = \{0\}$.

It can be checked that:

1. the series $\mathbb{P}_1$ is realized by the RTWA A_1;
2. the series $\mathbb{P}_2$ is realized by the RTWA A_2;
3. the series $\mathbb{P}_1 \cdot_a \mathbb{P}_2$ associates any tree t in L with the integer $2^{\mathrm{h}(t)}$, where $\mathrm{h}(t)$ is the height of t.

Since L is infinite, so is $\mathrm{Im}(\mathbb{P}_A)$. According to Lemma 5, $\mathbb{P}_1 \cdot_a \mathbb{P}_2$ can not be realized by any RWTA. ∎

Corollary 2. *Let Σ an alphabet and a be a symbol in Σ_0. The tree series realized by some RWTAs are not closed under a-product.*

The same reasoning can be applied on the case of iterated product.

3.5 Quotient of a RWTA

Morphisms of RWTAs can be applied w.r.t. an equivalence relation in order to define quotients of RWTA.

Given an equivalence relation $\sim$ over a set Q, we denote by $Q_\sim$ the set of equivalence classes of $\sim$. Given a state q in Q, we denote by $[q]_\sim$ the equivalence class of q w.r.t. $\sim$, *i.e.* $\{q' \in Q \mid q' \sim q\}$.

Definition 6. *Let $A = (\Sigma, Q, \nu, \delta)$ be a RWTA and $\sim$ be an equivalence relation over ν. The* quotient of A *w.r.t. $\sim$ is the RWTA $A_\sim = (\Sigma, Q_\sim, \nu', \delta')$ defined by:*

– $\forall C \in Q_\sim$, $\nu'(C) = \sum_{q \in C} \nu(C)$,
– $\forall C_1, \ldots, C_{k+1} \in Q_\sim$, $C_{k+1} \in \delta'(f, C_1, \ldots, C_k) \Leftrightarrow \forall i \leq k+1$, $\exists q_i \in C_i$, $q_{k+1} \in \delta(f, q_1, \ldots, q_k)$

Notice that the quotient of a RWTA A does not necessary realize the same series as A. Nevertheless, in the following of this paper, we use particular relation that preserves the series while quotienting.

Definition 7. *Let $A = (\Sigma, Q, \nu, \delta)$ be a RWTA and q be a state in Q. The* down language of q *is the language $L_q(A)$ defined by:*

$$L_q(A) = \{t \in T_\Sigma \mid q \in \Delta(t)\}.$$

Proposition 5. *The tree series realized by a RWTA $A = (\Sigma, Q, \nu, \delta)$ is equal to $\sum_{q \in Q} \nu(q) L_q(A)$.*

Proof. By definition, it holds that $\mathbb{P}_A = \sum_{t \in T_\Sigma} \nu(\Delta(t))t$. Consequently, by definition of $\nu(\Delta(t))$, $\mathbb{P}_A = \sum_{t \in T_\Sigma} \sum_{q \in \Delta(t)} \nu(q)t$. Furthermore, since any tree t such that $\Delta(t)$ is not empty is a tree that belongs to $L_q(t)$ for some state q in Q, $\mathbb{P}_A = \sum_{q \in Q} \sum_{t \mid q \in \Delta(t)} \nu(q)t$. Thus, by definition of $L_q(A)$, $\mathbb{P}_A = \sum_{q \in Q} \sum_{t \in L_q(A)} \nu(q)t$. Consequently, since the coefficient $\nu(q)$ belongs to a semiring, by distributivity, $\mathbb{P}_A = \sum_{q \in Q} \nu(q) L_q(A)$. ∎

Definition 8. *Let $A = (\Sigma, Q, \nu, \delta)$ be a RWTA. Let $\sim$ be an equivalence relation over Q. The relation $\sim$ is said to be* down compatible *with A if for any two states q_1 and q_2 in Q, it holds:*

$$q_1 \sim q_2 \Rightarrow L_{q_1}(A) = L_{q_2}(A).$$

Proposition 6. *Let A be a RWTA and $\sim$ be an equivalence relation down compatible with A. Then:*
$$\mathbb{P}_A = \mathbb{P}_{A_\sim}.$$

Proof. Let $A = (\Sigma, Q, \nu, \delta)$ and $A_\sim = (\Sigma, Q_\sim, \nu', \delta')$. According to Proposition 5, $\mathbb{P}_A = \sum_{q\in Q} \nu(q) L_q(A)$ and $\mathbb{P}_{A_\sim} = \sum_{C\in Q_\sim} \nu'(C) L_C(A)$. By definition of ν', $\mathbb{P}_{A_\sim} = \sum_{C\in Q_\sim} (\sum_{q\in C} \nu(q)) L_C(A)$. Since $\sim$ is down compatible, for any state C in $Q_\sim$, for any two states q_1 and q_2 in C, $L_{q_1}(A) = L_{Q_2}(A)$. Therefore $\mathbb{P}_{A_\sim} = \sum_{C\in Q_\sim} (\sum_{q\in C} \nu(q) L_q(A))$. Moreover, since $\sim$ is an equivalence relation, any state of Q belongs to one and only one state C in $Q_\sim$. Consequently, $\mathbb{P}_{A_\sim} = \sum_{q\in C} \nu(q) L_q(A) = \mathbb{P}_A$.

Example 5. Let us consider the RWTA $A'' = A \times A'$ represented in Figure 4. Let us consider the equivalence relation $\sim$ over the set of states of A'' defined by $(q_1, q_1') \sim (q_2, q_2') \Leftrightarrow q_1' = q_2'$. It can be shown that $\sim$ is down compatible with A. The quotient $A''_\sim$ is represented in Figure 5.

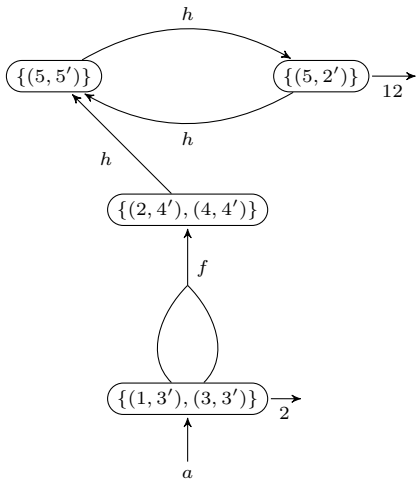

Fig. 5. The RWTA $A''_\sim$.

Now that we have defined the notion of RWTA, let us apply it on tree kernel computations.

4 Subtree Kernel

In this section, we show how to efficiently compute the subtree kernel of two finite tree languages using RWTAs. We first associate any tree with a RWTA that realizes its subtree series.

4.1 Subtree Automaton of a Tree

Definition 9. *Let Σ be an alphabet. Let t be a tree in T_Σ. The* subtree automaton *associated with t is the RWTA $A_t = (\Sigma, Q, \nu, \delta)$ defined by:*

- $Q = \mathrm{SubTreeSet}(t^\sharp)$,
- $\forall q \in Q$, $\nu(q) = 1$,
- $\forall f \in \Sigma_{t^\sharp}$, $\forall t_1, \ldots, t_{k+1} \in Q$, $t_{k+1} \in \delta(\mathrm{h}(f), t_1, \ldots, t_k) \Leftrightarrow t_{k+1} = f(t_1, \ldots, t_k)$.

Example 6. Let us consider the tree $t_1 = f(h(a), f(h(a), b))$ defined in Example 1. Then $t^\sharp = f_1(h_2(a_3), f_4(h_5(a_6), b_7))$. The RWTA A_{t_1} is represented in Figure 6, where all the root weights, equal to 1, are not represented.

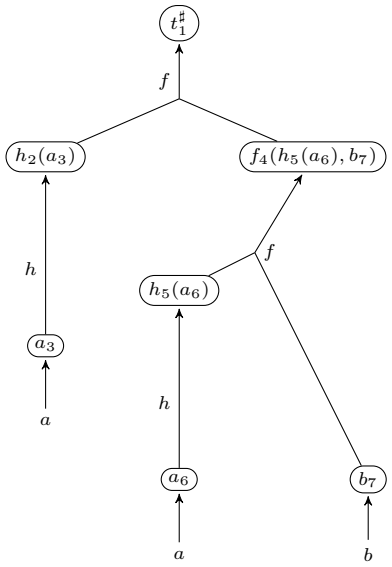

Fig. 6. The RWTA A_{t_1}.

Lemma 6. *Let Σ be an alphabet. Let t be a tree in T_Σ Then:*

$$\mathbb{P}_{A_t} = \text{SubTreeSeries}_t.$$

Proof. Let us set $A_t = (\Sigma, Q, \nu, \delta)$ and $A_{t_i} = (\Sigma, Q_i, \nu_i, \delta_i)$ for $1 \leq i \leq k$. Notice that by definition:

$$Q = \{t\} \cup \bigcup_{1 \leq i \leq k} Q_i \text{ and } \delta = \{(t, f, t_1, \ldots, t_k)\} \cup \bigcup_{1 \leq i \leq k} \delta_i.$$

Consequently, $\mathbb{P}_{A_t} = t + \sum_{1 \leq i \leq k} \mathbb{P}_{A_{t_i}}$. By definition, $\text{SubTreeSeries}_t = t + \sum_{1 \leq j \leq k} \text{SubTreeSeries}_{t_j}$. Furthermore, by induction hypothesis, $\mathbb{P}_{A_{t_i}} = \text{SubTreeSeries}_{t_i}$. Therefore it holds that

$$\mathbb{P}_{A_t} = t + \sum_{1 \leq j \leq k} \text{SubTreeSeries}_{t_j} = \text{SubTreeSeries}_t.$$

■

Another RWTA can be defined in order to realize the subtree series associated with a tree. This RWTA needs less space since its states are exactly its subsets.

Definition 10. *Let Σ be an alphabet. Let t be a tree in T_Σ. The* sequential subtree automaton *associated with t is the RWTA* $\text{seq}(A_t) = (\Sigma, Q, \nu, \delta)$ *defined by:*

- $Q = \text{SubTreeSet}(t)$,
- $\forall t' \in Q$, $\nu(t') = \text{SubTreeSeries}_t(t')$,
- $\forall f \in \Sigma$, $\forall t_1, \ldots, t_{k+1} \in Q$, $t_{k+1} \in \delta(f, t_1, \ldots, t_k) \Leftrightarrow t_{k+1} = f(t_1, \ldots, t_k)$.

Example 7. Let us consider the tree $t_1 = f(h(a), f(h(a), b))$ defined in Example 1. The RWTA $\text{seq}(A_{t_1})$ is represented in Figure 7.

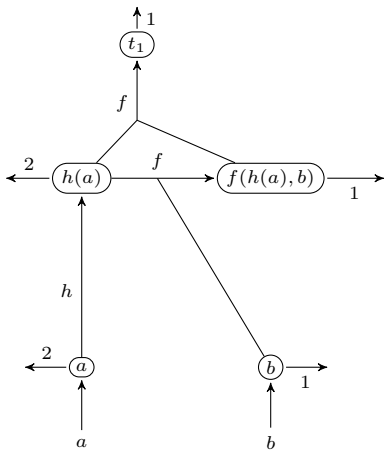

Fig. 7. The RWTA $\mathrm{seq}(A_{t_1})$.

However, the sequential subtree automaton needs the tree series to be known in order to compute it. Nevertheless, we show how to compute it from a quotient of the subtree automaton. Once this tree computed, it can be reduced using the equivalence $\sim_{\mathrm{h}}$. Furthermore, this RWTA is isomorphic to the one obtained by subset construction. Consequently, we compute a sequential RWTA the number of states oh which is equal to the number of its different subtrees. We first show that $\sim_{\mathrm{h}}$ is down compatible with the subtree automaton, then we show that its application leads to the computation of the sequential subtree automaton.

Lemma 7. *Let Σ be a graded alphabet. Let t be a tree in T_Σ. Let $A_t = (\Sigma, Q, \nu, \delta)$. For any tree r in* $\mathrm{SubTreeSet}(t)$*, it holds:*

$$\Delta(r) = \{r' \in Q \mid \mathrm{h}(r') = r\}.$$

Proof. By induction over the structure of $r = f(r_1, \ldots, r_k)$. By definition of Δ, $\Delta(r) = \delta(f, \Delta(r_1), \ldots, \Delta(r_k))$. By induction hypothesis, $\Delta(r_i) = \{r'_i \in Q \mid \mathrm{r}'_\mathrm{i} = r_i\}$. Thus, $\Delta(r) = \{f_j(r'_1, \ldots, r'_k) \in Q \mid \mathrm{h}(f_j) = f \wedge r_i = \mathrm{h}(r'_i)\}$. Hence $\Delta(r) = \{r' \in Q \mid \mathrm{h}(r') = r\}$. ∎

As a direct consequence, for any state r in Q, any tree r' in $L_r(A_t)$ satisfies $\mathrm{h}(r) = r'$.

Corollary 3. *Let Σ be a graded alphabet. Let t be a tree in T_Σ. Let $A_t = (\Sigma, Q, \nu, \delta)$. Then for any state r in Q, $L_r(A_t) = \{\mathrm{h}(r)\}$.*

Lemma 8. *Let Σ be a graded alphabet. Let t be a tree in T_Σ. Then:*
$\sim_{\mathrm{h}}$ *is down compatible with A_t.*

Proof. Let $A_t = (\Sigma, Q, \nu, \delta)$. According to Corollary 3, $L_r(A_t) = \{\mathrm{h}(r)\}$. Consequently, for any two states r_1 and r_2 in Q, $r_1 \sim_{\mathrm{h}} r_2 \Rightarrow \mathrm{h}(r_1) = \mathrm{h}(r_2) \Rightarrow L_{r_1}(A_t) = L_{r_2}(A_t)$. ∎

Proposition 7. *Let Σ be a graded alphabet. Let t be a tree in T_Σ. Then:*
The RWTA $\mathrm{seq}(A_t)$ *is isomorphic to $A_{t_{\sim_h}}$.*

Proof. Let us set $A_t = (\Sigma, Q = \mathrm{SubTreeSet}(t^\sharp), \nu, \delta)$, $A_{t_{\sim_h}} = (\Sigma, Q_\sim, \nu', \delta')$ and $\mathrm{seq}(A_t) = (\Sigma, \mathrm{SubTreeSet}(t), \nu'', \delta'')$.

By definition, any state $C = \{t_1, \ldots\}$ in $Q_{\sim_h}$ can be associated with $\mathrm{h}(t_1)$ since any two states q and q' in C satisfies by definition $\mathrm{h}(q) = \mathrm{h}(q')$. Consequently, let us consider the function g that associates to any state $C = \{t_1, \ldots, \}$ in $Q_{\sim_h}$ the tree $\mathrm{h}(t_1)$. Notice that this function is bijective since for any tree r, $g^{-1}(r) = \{r' \in \mathrm{SubTreeSet}(\mathrm{t}^\sharp) \mid \mathrm{h}(\mathrm{r}') = \mathrm{r}\}$. Let us show that this function defines an isomorphism between $A_{t_{\sim_h}}$ and $\mathrm{seq}(A_t)$.

1. By definition, for any state C in $Q_{\sim_h}$, $g(f)$ belongs to $\mathrm{SubTreeSet}(t)$.

2. For any transition $(C_{k+1}, f, C_1, \ldots, C_k)$ in δ', there exist by definition $t_1, \ldots, t_k$ in Q and a symbol f_j satisfying $\mathrm{h}(f_j) = f$ such that $(f_j(t_1, \ldots, t_k), f, t_1, \ldots, t_k)$ is in δ. Consequently $f_j(t_1, \ldots, t_k)$ is a subtree of $t^\sharp$ and then $f(\mathrm{h}(t_1), \ldots, \mathrm{h}(t_k))$ is a subtree of t. Therefore by definition of A $(g(C_{k+1}) = \mathrm{h}(t_{k+1}), f, g(C_1) = \mathrm{h}(t_1), \ldots, g(C_k) = \mathrm{h}(t_k))$ is in δ''.
3. According to Corollary 3, for any state r in Q, $L_r(A_t) = \{\mathrm{h}(r)\}$. Consequently, for any state C in $Q_\sim$, $\nu'(C) = \sum_{r \in C} \nu(r)$. According to Lemma 7, for any tree r', $\Delta(r') = \{r \mid \mathrm{r} = r'\} = C$. Moreover, according to Lemma 6, for any tree r', $\nu(\Delta(r')) = \mathrm{SubTreeSeries}_t(r')$. Then $\nu'(C) = \mathrm{SubTreeSeries}_t(g(C)) = \nu''(g(C))$.

■

We denote by $|t|$ the size of a tree t, *i.e.* the number of its nodes. Since the subtree automaton and the relation $\sim_\mathrm{h}$ can be computed in linear time, it holds that

Corollary 4. *Let Σ be an alphabet. Let t be a tree in T_Σ. Then:*
The RWTA $\mathrm{seq}(A_t)$ *can be computed in time and space $O(|t|)$.*

Let us now show that the sequential subtree automaton is a sequential RWTA. Let us prove it by showing that the computation of the accessible part of the subset construction leads exactly to the computation of the quotient of the subtree automaton, where the *accessible part* of the sequential RWTA associated with a RWTA is the RWTA based on the states the down languages of which is not empty.

Proposition 8. *Let Σ be an alphabet. Let t be a tree in T_Σ. Then:*
The accessible part of the sequential RWTA associated with A_t is equal to $A_{t\sim_h}$.

Proof. Let us set $A_t = (\Sigma, Q = \mathrm{SubTreeSet}(t^\sharp), \nu, \delta)$, $A' = (\Sigma, Q', \nu', \delta')$ the accessible part of sequential RWTA associated with A_t and $A'' = A_{t\sim_h} = (\Sigma, Q'', \nu'', \delta'')$.

According to Lemma 3, $\Delta'(r') = \{\Delta(r')\}$. According to Lemma 7, $\Delta(r') = \{r \in \mathrm{SubTreeSet}(t^\sharp) \mid \mathrm{h}(r) = r'\}$. Hence, $\Delta'(r') = \{\{r \in \mathrm{SubTreeSet}(t^\sharp) \mid \mathrm{h}(r) = r'\}\}$ that is an equivalence class of $\sim_\mathrm{h}$. Therefore, $Q'' = Q'$ and $\delta' = \delta''$. Moreover, for any state C in Q', $\nu'(C)$ and $\nu''(C)$ are both equal by definition to $\sum_{c \in C} \nu(c)$. Consequently $A' = A''$. ■

Corollary 5. *Let Σ be an alphabet. Let t be a tree in T_Σ. Then:*
The accessible part of the sequential RWTA associated with A_t is isomorphic to $\mathrm{seq}(A_t)$.

To sum up the properties of the sequential subtree RWTA associated with a tree:

Corollary 6. *Let Σ be an alphabet. Let t be a tree in T_Σ. Then the RWTA* $\mathrm{seq}(A_t)$*:*

- *is a sequential RWTA,*
- *is smaller than A_t,*
- *realizes* $\mathrm{SubTreeSeries}_t$*,*
- *is constructed in time and space $O(|t|)$.*

Let us now show how to extend this construction to finite tree languages.

4.2 Subtree Automaton of a Finite Tree Language

Let us first define a RWTA recognizing the subtree series of a finite tree languages.

Definition 11. *Let Σ be an alphabet. Let L be a finite tree language over Σ. The* sequential subtree automaton *associated with L is the RWTA $A_L = (\Sigma, Q, \nu, \delta)$ defined by:*

- $Q = \mathrm{SubTreeSet}(L)$,
- $\forall t' \in Q$, $\nu(t') = \mathrm{SubTreeSeries}_L(t')$,
- $\forall f \in \Sigma$, $\forall t_1, \ldots, t_{k+1} \in Q$, $t_{k+1} \in \delta(f, t_1, \ldots, t_k) \Leftrightarrow t_{k+1} = f(t_1, \ldots, t_k)$.

Notice that by definition, for any tree t, $\mathrm{seq}(A_t)$ and $A_{\{t\}}$ are isomorphic.

Example 8. Let us consider the trees $t_1 = f(h(a), f(h(a), b))$ and $t_2 = f(h(a), h(b))$ defined in Example 1. The RWTA $A_{\{t_1,t_2\}}$ is represented in Figure 8 and realized the series $\text{SubTreeSeries}_{\{t_1,t_2\}} = t + t_2 + f(h(a), b) + 3h(a) + h(b) + 3a + 2b$.

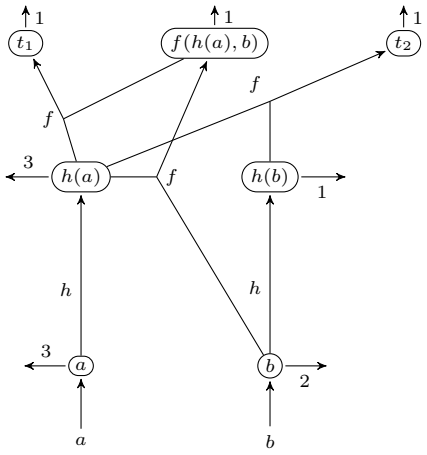

Fig. 8. The RWTA $A_{\{t_1,t_2\}}$.

Similarly to the case of the sequential subtree RWTA of a tree, the subtree RWTA A of a language needs the subtree series to be *a priori* known. Let us show that A can be computed without knowing the series. In order to compute it, we make use of the sum and of the sequentialization, two operations defined in Section 3.

Lemma 9. *Let Σ be an alphabet. Let L be a finite tree language over Σ. Let $A_L = (\Sigma, Q, \nu, \delta)$. For any tree r in* $\text{SubTreeSet}(L)$*, it holds:*

$$\Delta(r) = \{r\}.$$

Proof. By induction over the structure of $r = f(r_1, \ldots, r_k)$. By definition of Δ, $\Delta(r) = \delta(f, \Delta(r_1), \ldots, \Delta(r_k))$. By induction hypothesis, $\Delta(r_i) = \{r_i\}$. Thus, $\Delta(r) = \delta(f, r_1, \ldots, r_k)$. Hence $\Delta(r) = \{r\}$. ∎

As a direct consequence of Lemma 9, A_L is sequential. Let us show now that this RWTA can be obtained by an inductive sequentialization.

Proposition 9. *Let Σ be an alphabet. Let L_1 and L_2 be two distinct finite tree languages over Σ. Then: The RWTA $A_{L_1 \cup L_2}$ is isomorphic to the accessible part of the sequential RWTA associated with $A_{L_1} + A_{L_2}$.*

Proof. By recurrence over the cardinality of L_2.

1. If L_2 is empty, then the proposition is satisfied.
2. Suppose that $L_2 = L_2' \cup \{t\}$ with $t \notin L_2'$. Then according to the recurrence hypothesis, the RWTA $A' = A_{L_1 \cup L_2'} = (\Sigma, Q', \nu', \delta')$ is isomorphic to the sequential RWTA associated with $A_{L_1} + A_{L_2'}$. Let $A_{\{t\}} = (\Sigma, Q_t, \nu_t, \delta_t)$, $A'' = A' + A_{\{t\}} = (\Sigma, Q'', \nu'', \delta'')$ and $A''' = (\Sigma, Q''', \nu''', \delta''')$ be the sequential RWTA associated with A''. By construction, either $Q' \cap \text{SubTreeSet}(t)$ is empty, or the states r of $A_{\{t\}}$ have to be relabelled as $\overline{r}$.
   (a) By construction, if $Q' \cap \text{SubTreeSet}(t)$ is empty, it holds from Lemma 7 and from Lemma 9 that the construction of the accessible part of the sequential RWTA associated with A'' is just a relabelling of the states. Furthermore, it can be shown by definition of A'' that $A'' = A_{L_1 \cup L_2}$. Hence $A_{L_1 \cup L_2}$ is isomorphic to the acecessible part of the sequential RWTA associated with $A_{L_1} + A_{L_2}$.
   (b) Otherwise, according to Lemma 3, $\Delta'''(r) = \{\Delta''(r)\}$, that equals by construction to the set $\{\Delta_t(r) \cup \Delta'(r)\}$. Hence the states of the accessible part of A''' are $\text{SubtreeSet}(L_1 \cup L_2') \cup \text{SubtreeSet}(t)$ that equals by definition to $\text{SubtreeSet}(L_1 \cup L_2)$. Furthermore, for any state r in Q''',

$$\nu'''(\Delta'''(r)) = \nu''(\Delta''(r)) = \begin{cases} \nu''(\{r,\overline{r}\}) = \nu_t(r) + \nu'(r) & \text{if } r \in Q' \cap \text{SubTreeSet}(t), \\ \nu''(\{r\}) = \nu'(r) & \text{if } r \in Q' \setminus \text{SubTreeSet}(t), \\ \nu''(\{\overline{r}\}) = \nu_t(\overline{r}) & \text{if } r \in \text{SubTreeSet}(t) \setminus Q'. \end{cases}$$

Consequently, since A''' is sequential, for any state r in Q''',
$\nu'''(r) = \nu'''(\Delta'''(r)) = \text{SubTreeSeries}_{L_1 \cup L_2'}(r) + \text{SubTreeSeries}_t(r) = \text{SubTreeSeries}_{L_1 \cup L_2}(r)$.
Finally, since by construction of A''', $\forall f \in \Sigma$, $\forall \{t_1\}, \ldots, \{t_{k+1}\} \in Q'''$, $\{t_{k+1}\} \in \delta(f, \{t_1\}, \ldots, \{t_k\})$
$\Leftrightarrow t_{k+1} = f(t_1, \ldots, t_k)$, it holds that A''' is isomorphic to $A_{L_1 \cup L_2}$.

■

Notice that since the complexity of the computation of the accessible part of $A_{L_1} + A_{L_2}$ is equal to the size of $A_{L_1 \cup L_2}$, by setting for any tree language L, $|L| = \sum_{t \in L} |t|$, it can be performed in time equal to $|L_1| + |L_2|$, and not in an exponential time. Moreover, as a direct consequence of Proposition 1, Proposition 2, Proposition 5 and Proposition 9

Corollary 7. *Let Σ be an alphabet. Let L be a finite tree language over Σ. Then:*
The RWTA A_L is a sequential RWTA that realizes SubTreeSeries_L.

Consequently the subtree automaton A_L associated with a finite tree language L can be computed by summing and sequentializing all the $A_{\{t\}} \equiv A_t$ for t in L. Therefore, since the coomputation of the sequential subtree RWTA of any tree, the sum and the sequentialization (in this case) can be computed in linear time, it holds that, :

Corollary 8. *Let Σ be an alphabet. Let L be a finite tree language over Σ. Then:*
The RWTA A_L can be computed in time $O(|L|)$.

4.3 Kernel Computation

In order to compute the subset kernel of two finite tree languages L_1 and L_2, we first compute the two RWTAs A_{L_1} and A_{L_2}; Then we compute the cartesian product $A_{L_1} \times A_{L_2}$; Finally we sum all the root weight of this RWTA.

Let us first show that our *modus operandi* is correct:

Theorem 2. *Let Σ be an alphabet. Let L_1 and L_2 be two finite tree languages over Σ. Let (Σ, Q, ν, δ) be the accessible part of $A_{L_1} \times A_{L_2}$. Then*

$$\text{KerSeries}(L_1, L_2) = \nu(Q).$$

Proof. Let us set $A_{L_1} = (\Sigma, Q_1, \nu_1, \delta_1)$ and $A_{L_2} = (\Sigma, Q_2, \nu_2, \delta_2)$.

By definition of the series product and from Corollary 7,

$$\begin{aligned} \text{KerSeries}(L_1, L_2) &= \textstyle\sum_{t \in T_\Sigma} (\text{SubTreeSeries}_{L_1} \times \text{SubTreeSeries}_{L_2})(t) \\ &= \textstyle\sum_{t \in T_\Sigma} (\text{SubTreeSeries}_{L_1}(t) \times \text{SubTreeSeries}_{L_2}(t)) \\ &= \textstyle\sum_{t \in T_\Sigma} (\nu_1(t) \times \nu_2(t)). \end{aligned}$$

According to Lemma 9, for any tree t in T_Σ, $\Delta_1(t)$ (resp. $\Delta_2(t)$) is either equal to $\{t\}$ if $t \in \text{SubTreeSet}(L_1)$ ($t \in \text{SubTreeSet}(L_2)$) or to $\emptyset$. Therefore, according to Lemma 4, $\Delta(t)$ is either equal to $\{(t,t)\}$ if $t \in \text{SubTreeSet}(L_1) \cap \text{SubTreeSet}(L_2)$, $\emptyset$ otherwise. Moreover, by definition of ν, for any tree t, $\nu(t) = \nu(\Delta(t)) = \nu_1(t) \times \nu_2(t)$. Furthermore, by definition of Q, t is in Q if and only if $t \in \text{SubTreeSet}(L_1) \cap \text{SubTreeSet}(L_2)$. Consequently, $\nu(Q) = \sum_{t \in \text{SubTreeSet}(L_1) \cap \text{SubTreeSet}(L_2)} \nu_1(t) \times \nu_2(t)$. Since for any tree t, if $t \notin \text{SubTreeSet}(L_1)$ (resp. $t \notin \text{SubTreeSet}(L_2)$), then $\nu_1(t) = 0$ (resp. $\nu_2(t) = 0$), it holds that

$$\nu(Q) = \textstyle\sum_{t \in T_\Sigma} \nu_1(t) \times \nu_2(t) = \text{KerSeries}(L_1, L_2).$$

■

Finally, by combining the elemental complexities,

Theorem 3. *Let Σ be an alphabet. Let L_1 and L_2 be two finite tree languages over Σ. Then*
$\text{KerSeries}(L_1, L_2)$ *can be computed in time* $O(|L_1| + |L_2| + \text{Card}(\text{SubTreeSet}(L_1) \cap \text{SubTreeSet}(L_2)))$.

Proof. From Corollary 8, $A_{L_1} = (\Sigma, Q_1, \nu_1, \delta_1)$ and $A_{L_2} = (\Sigma, Q_2, \nu_2, \delta_2)$ are constructed in time $O(|L_1| + |L_2|)$. From Lemma 4, the accessible part of $A_{L_1} \times A_{L_2}$ is composed of the set S of states of the form (t, t) with $t \in \mathrm{SubTreeSet}(L_1) \cap \mathrm{SubTreeSet}(L_2)$. From Theorem 2, $\mathrm{KerSeries}(L_1, L_2)$ is computed summing the root weight of the states in S. Therefore $\mathrm{KerSeries}(L_1, L_2)$ can be computed in time $O(|L_1| + |L_2| + \mathrm{Card}(\mathrm{SubTreeSet}(L_1) \cap \mathrm{SubTreeSet}(L_2)))$. ∎

Example 9. Let us consider the trees $t_1 = f(h(a), f(h(a), b))$, $t_2 = f(h(a), h(b))$ and $t_3 = f(f(b, h(b)), f(h(a), h(b)))$ defined in Example 1. The RWTA $A_{\{t_3\}}$ is represented in Figure 9. The RWTA $R = A_{\{t_1,t_2\}} \times A_{\{t_3\}}$ is represented in Figure 10. The sum of the root weights of R is equal to 15, that is $\mathrm{KerSeries}(\{t_1, t_2\}, \{t_3\})$.

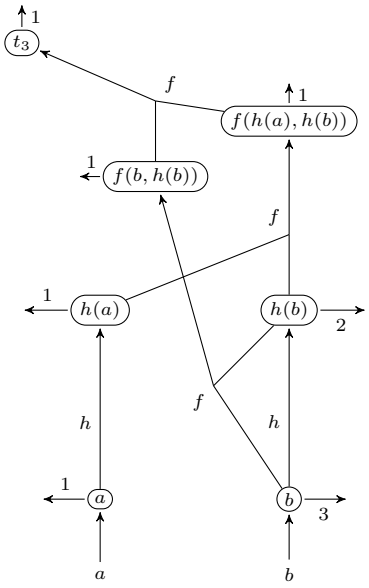

Fig. 9. The RWTA $A_{\{t_3\}}$.

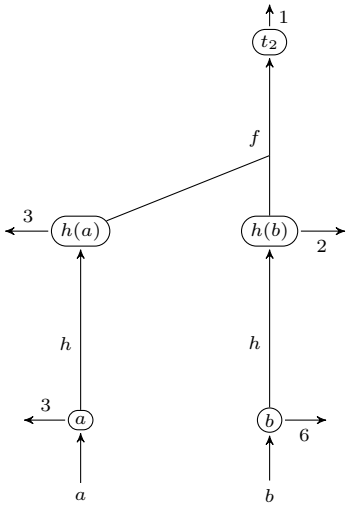

Fig. 10. The RWTA $A_{\{t_1,t_2\}} \times A_{\{t_3\}}$.

5 Conclusion and Perspectives

In this paper, we defined new weighted tree automata that are always sequentializable but that does not realize all the classical recognizable series. We studied the different algebraic combinations of these automata (sum, products, regular operations) in order to determine their closures. Once these definitions stated, we made use of these new structures in order to compute the subtree kernel of two finite tree series in an efficient way.

Our technique can be applied to other computations. Indeed, other tree kernels exist, like the SST kernel. The next step of our work is to apply our constructions in order to efficiently compute these kernels. However, this application is not so direct since it seems that the SST series may not be sequentializable w.r.t. a linear space complexity. Hence we have to find different techniques, like extension of lookahead determinism [11] for example.

Another perspective is related to the series realized by RWTAs. It is an open question to determine what family they exactly are.

References

1. Aizerman, M.A., Braverman, E.M., Rozonoèr, L.I.: Theoretical foundation of potential functions method in pattern recognition. Avtomat. i Telemekh. **25**(6) (1964) 917–936
2. Berstel, J., Reutenauer, C.: Recognizable formal power series on trees. Theor. Comput. Sci. **18** (1982) 115–148

3. Collins, M., Duffy, N.: New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA. (2002) 263–270
4. Comon, H., Dauchet, M., Gilleron, R., Jacquemard, F., Lugiez, D., Loding, C., Tison, S., Tommasi, M.: Tree automata techniques and applications. Available on: `http://www.grappa.univ-lille3.fr/tata` (October 2007)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3) (1995) 273–297
6. Culotta, A., Sorensen, J.S.: Dependency tree kernels for relation extraction. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain. (2004) 423–429
7. Cumby, C.M., Roth, D.: On kernel methods for relational learning. In: Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA. (2003) 107–114
8. Droste, M., Pech, C., Vogler, H.: A kleene theorem for weighted tree automata. Theory Comput. Syst. **38**(1) (2005) 1–38
9. Ésik, Z., Kuich, W.: Formal tree series. Journal of Automata, Languages and Combinatorics **8**(2) (2003) 219–285
10. Fülöp, Z., Maletti, A., Vogler, H.: A kleene theorem for weighted tree automata over distributive multioperator monoids. Theory Comput. Syst. **44**(3) (2009) 455–499
11. Han, Y., Wood, D.: Generalizations of 1-deterministic regular languages. Inf. Comput. **206**(9-10) (2008) 1117–1125
12. Moschitti, A.: A study on convolution kernels for shallow statistic parsing. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain. (2004) 335–342
13. Moschitti, A.: Efficient convolution kernels for dependency and constituent syntactic trees. In: Machine Learning: ECML 2006, 17th European Conference on Machine Learning, Berlin, Germany, September 18-22, 2006, Proceedings. (2006) 318–329
14. Moschitti, A.: Making tree kernels practical for natural language learning. In: EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, April 3-7, 2006, Trento, Italy. (2006) 113–120
15. Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM J. Res. **3**(2) (1959) 115–125
16. Schölkopf, B., Smola, A.J., Müller, K.: Kernel principal component analysis. In: Artificial Neural Networks - ICANN '97, 7th International Conference, Lausanne, Switzerland, October 8-10, 1997, Proceedings. (1997) 583–588
17. Zelenko, D., Aone, C., Richardella, A.: Kernel methods for relation extraction. Journal of Machine Learning Research **3** (2003) 1083–1106